Information modelling: foundation, abstraction mechanisms and approach

Jørgensen, Kaj Asbjørn

Published in:
Journal of Intelligent Manufacturing

Publication date:
1998

Document Version
Accepted author manuscript, peer reviewed version

Aalborg University – Department of Production

2010-02-09

Information Modelling: foundation, abstraction mechanisms and approach

Kaj A. Jørgensen

Abstract: This paper briefly presents a new information modelling methodology useful as an information systems development approach. The methodology is based on a set of fundamental abstraction mechanisms, which provide the means for modelling on a number of abstraction levels and, on the highest levels, the most invariant components and structures of the information model can be identified and defined. It is also based on the object-oriented paradigm, which is applied in harmony with the abstraction mechanisms. The methodology introduces an alternative to the traditional entity-relationship technique with respect to modelling of information and on this basis it can support efforts to construct *reusable*, *extensible* and *reliable* information systems.

1 Information Modelling as the Basis for Information Systems Development

Fundamentally, an information system is developed by creating an *object base* and a set of *user interfaces.* The object base contains the persistent information objects and the user interfaces provide the functionality, which can be operated by the users of the information system. For development of information systems, some different kinds of software tools are now available to support the development activities. With some tools, information systems are developed through a modelling process by creating and manipulating information system models from which the resulting system can

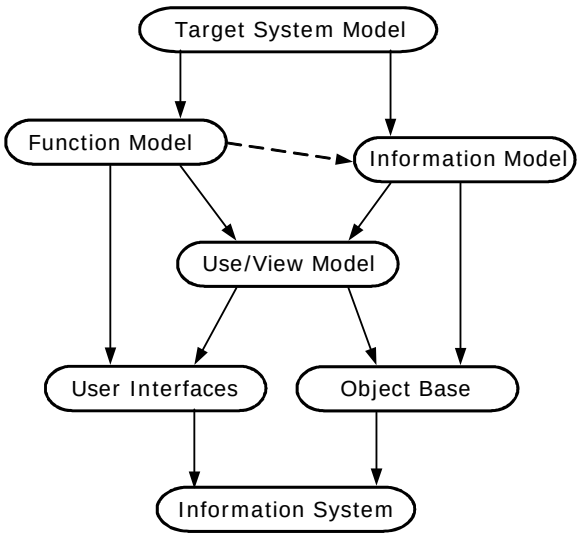

Figure 1 - The components of information system development by modelling

be generated. Many tools have flexible graphical design interfaces, detailed repositories, comprehensive facilities for generation of object bases and user interfaces, etc. In order to advance from this situation, a new information modelling methodology is developed and presented in the following. Figure 1 shows the components, which are included in information system development projects.

Each information system is developed in order to be implemented and function in some kind of real world system termed *the target system* and a model of such a system, e.g. a management system, is assumed to be the starting point of the development process (figure 1). From this model, the overall requirements to the information system are derived as the basis for modelling. Fundamentally, two models are necessary: *function model* and an *information model*. The function model is a model of the operations, which the information system must provide for the users and the information model is a model of the information which is handled by the tasks.

Different approaches can be followed for modelling and different importance can be put on the two models. In a *process driven approach*, the function model is regarded as the most important and this model is usually specified down to a great detail. In a *data driven approach*, the information model is regarded as the most important. The advantage of this approach is primarily that an information model is relatively *invariant* against changes in the functional requirements. Besides, if object-orientation is applied, modelling of functionality is, to a large extent, included in information modelling and many user interfaces can be generated just from the information model.

The *use model* or *view model* is the model in which specific uses of the function model are related to specific components (views) of the information model. Each view can be detailed regarding the needs for *select*, *insert*, *update* and *delete* from the object base. The object base is generated primarily from the information model. In the following, the focus is on information modelling.

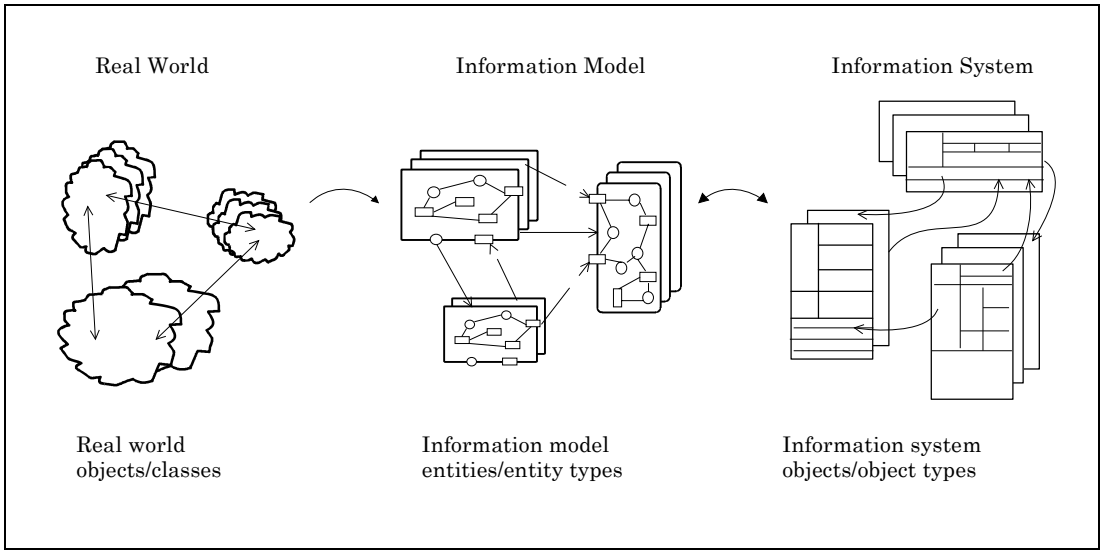

Figure 2 - Information Modelling

The information model may be considered as a link between the real world and the information system (figure 2). Seen from one side, it defines a view of the real world formed by a set of decisions which are based on the designers' communications with the participants of the project, e.g. contractors, supervisors, end-users, etc. ([Hammer 1978]) Seen from the other side, the model serves as a platform for specification and implementation of the information system. Therefore, the model will also contain detailed specifications aimed at the construction of the information system.

According to the object-oriented paradigm, the *attributes* of objects are divided into *factual attributes* and *behavioural attributes*, of which behavioural attributes are defined as relevant operations to be performed on the factual attributes. In addition, attributes can be specified as *visible* or *hidden* attributes. Visible attributes are accessible from outside the object, whereas hidden attributes are not directly accessible. They are only accessible via the behavioural attributes. Hidden attributes are

encapsulated in the object (figure 3). Visible attributes are sometimes termed *public* attributes while hidden attributes are termed *private*. The set of visible attributes, as a whole, is termed the *form* or the *interface* defining the *services* provided by the object. Recursively, objects can contain structures of internal objects (figure 3).

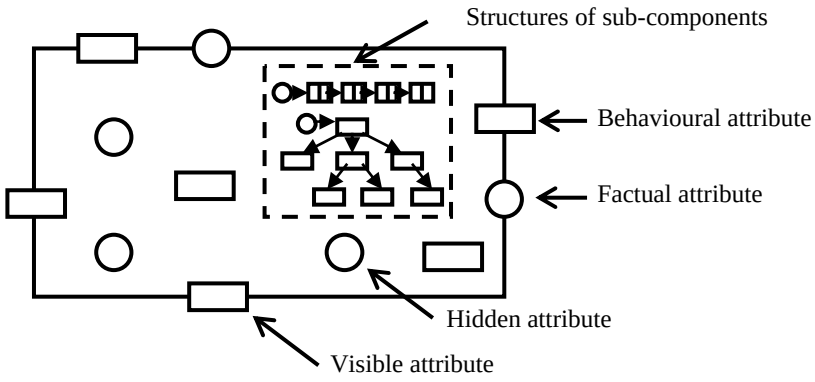

Figure 3 - Attributes of objects - visible and hidden attributes - internal objects.

The specifications of objects are made in *object types*. In order to distinguish object types from objects in diagrams, types are shown as a rounded boxes (figure 4). From each object type, an indefinite number of objects, instances, can be generated; see figure 5.

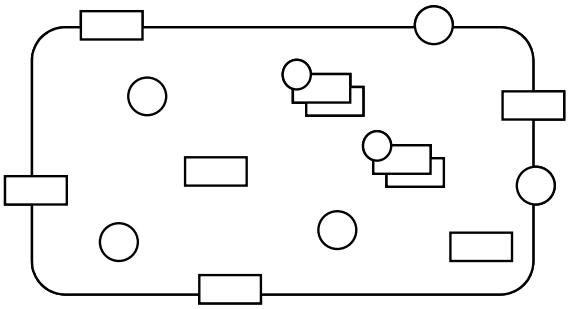

Figure 4 - Object type.

In order to distinguish clearly, the information model components are termed *entities* and *entity types*. Thus to one side, *information model entities* correspond to *real world objects*, and to the other side, they correspond to *information system objects*.

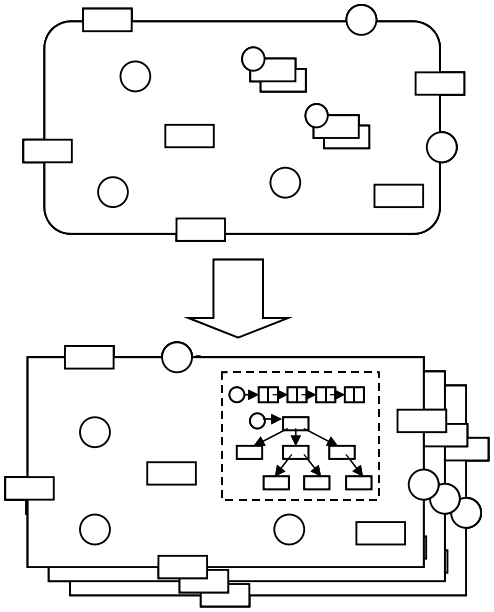

Figure 5 – Objects can be generated from object types.

It is considered a general requirement to a modelling process that the main decisions should be sequenced according to how invariant they are, implying that the decisions should be taken on different abstraction levels so that the most invariant decisions are taken on the top level and that decisions at one level restrict the decisions to be taken at the next level, etc. This requirement is stated as a primary design criterion for the methodology. In addition, the methodology should encourage *participation* and *communication* and it should enable *productive design processes* by which *reusable, extensible* and *reliable* information systems can be constructed.

2 Abstraction Mechanisms and Information Modelling

The information modelling methodology is developed partly on the basis of the framework of Avison [Avison 1988]. Accordingly, it is based on a defined *science paradigm;* it is related to a set of *design layers* of the development process and it is solidly based on a set of fundamental *abstraction mechanisms* ([Smith 1977a], [Smith 1977b], [Rosch 1978] and [Sowa 1984]), which provide the means for identification and design of different invariant components and structures. In addition, it is based on the *object-oriented paradigm,* which is applied in harmony with the abstraction mechanisms and, conforming to the traditional *three level design paradigm*, the methodology relates to the *conceptual* and *logical* levels, and the resulting information model can be used as a basis for *physical* design and *implementation.*

The methodology is based on two abstraction mechanisms: *composition* and *classification* which are defined on the basis of the fundamental concepts: *analysis* and *synthesis* known from general systems theory ([Boulding 1956] and [Bertalanffy 1967]). In essence, composition is the development of a hierarchy of systems determined by their components and classification is the development of a hierarchy of systems - the *taxonomy* - determined by the attributes of the systems. These mechanisms cover modelling on both the *type level* and the *instance level* of the information model and they provide the means to set particular focus on decisions related to the type level as the most invariant decisions. Each of the abstraction mechanisms is characterised by two complementary operations: in classification: *generalisation* versus *specialisation* and in composition: *aggregation* versus *separation.*

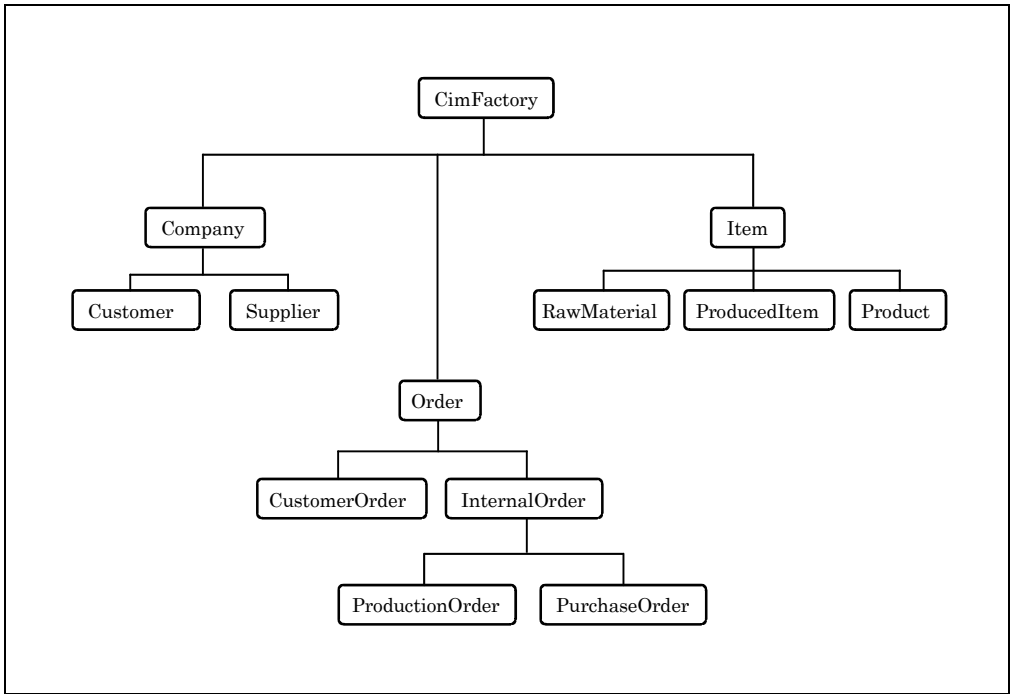

Figure 6 - A taxonomy of entity types

2.1 Classification of Entity Types

On the type level of the information model, the basic components, the entity types, are identified and defined. This is primarily performed by means of the classification abstraction mechanism and the entity types are organised in a taxonomy (figure 6). In the taxonomy, the most general types are placed at the higher levels and the most special types are placed at the lower levels according to the specified *attributes* (table 1). By adding and subtracting attributes to entity types, important abstraction processes will be performed in relation to the taxonomy, which provides a precise background for the operations.

ITEM - ITEMNAME - WEIGHT - INVENTORYQTY	PRODUCT - ITEMNAME - WEIGHT - INVENTORYQTY - SALESPRICE - COLOUR
PRODUCEDITEM - ITEMNAME - WEIGHT - INVENTORYQTY - MATERIALCOST - PROCESSINGCOST - PROCESSINGTIME	RAWMATERIAL - ITEMNAME - WEIGHT - INVENTORYQTY - MATERIAL - UNITOFMEASURE - PURCHASEPRICE

Table 1 - Sample attributes of four entity types

According to object-orientation, attributes of super-types are *inherited* to sub-types. This means that each entity type adds to its own list of attributes the attributes of all its ancestor-types (all super-types in the path from the root type to the concerned type). So for each entity type, it is only necessary to define and specify the local attributes, the attributes, which distinguish the entity type from its super-types (table 2). A number of semantic restrictions concerning uniqueness, initiation, integrity, etc. can be defined for entity types.

ITEM inherits from CIMFACTORY - ITEMNAME - WEIGHT - INVENTORYQTY	PRODUCT inherits from ITEM - SALESPRICE - COLOUR
PRODUCEDITEM inherits from ITEM - MATERIALCOST - PROCESSINGCOST - PROCESSINGTIME	RAWMATERIAL inherits from ITEM - MATERIAL - UNITOFMEASURE - PURCHASEPRICE

Table 2 - Local attributes of entity types – inheritance is specified

2.2 Composition of Entity Types

The composition abstraction mechanism is used to identify internal components of entities, in addition to the attributes. These components are also defined as entities and they are related to each other in internal structures. An entity, which has internal entities, is termed a *composite entity*. The internal entities are termed *member entities* and the composite entity is termed the *owner entity*. In such an

entity, the variable number of internal entities is organised in one or more internal structures, which are anchored to the composite entity. Each composite entity type can be regarded as a frame in which an *internal information model* with an *internal taxonomy* can be defined (figure 7). The introduction of composite entity types adds another dimension to the modelling approach and enables *recursive modelling of multi-level complexity*. It provides the designers an extra degree of flexibility in relation to stepwise refinement of the model; it can be build in steps by adding further details both to the primary taxonomy and to the internal taxonomies of composite entity types (figure 8).

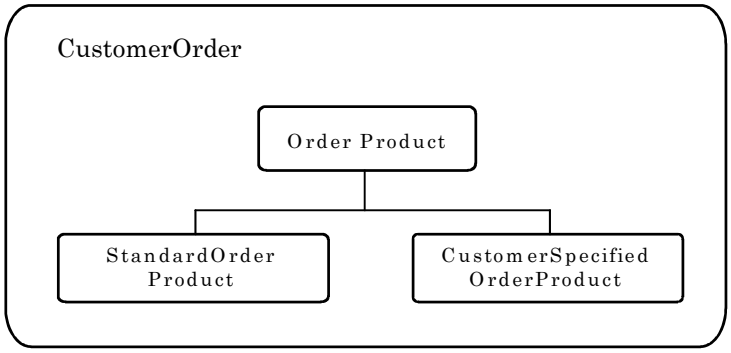

Figure 7 - An internal taxonomy of a composite entity type

During classification, it has to be decided in what direction a taxonomy should be developed. Simple information models may contain only one taxonomy. If an entity type is changed to a composite entity type, some entity types of the main taxonomy may be moved internal to composite entity types and, thereby, the number of entity types in the main taxonomy will be reduced. In contrast, if a composite entity type is changed to a non-composite entity type, the internal taxonomy must be added to the main taxonomy.

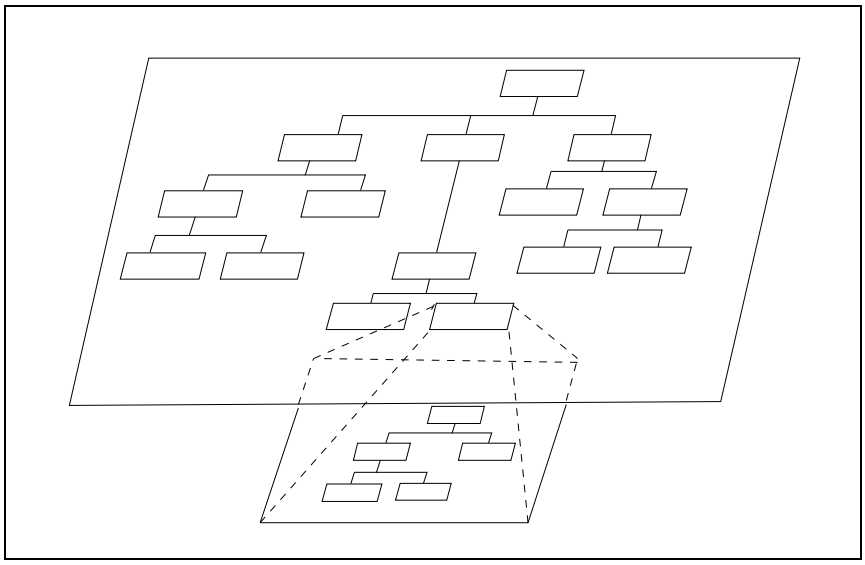

Figure 8 - Composite entity type with internal taxonomy

An internal taxonomy of a composite entity type is inherited to sub-types as a *template*, which can be extended by further entity types and attributes. This means that a new dimension is added to classification and, consequently, the overall structure of an information model will be a *hierarchy of taxonomies.* In the modelling process, it is necessary to decide where to put the entity types and, by the possibility of inheriting taxonomies as templates from a composite entity type to its sub-types, a further flexibility is provided in the modelling process.

2.3 Relationship Types

Besides the structures already mentioned, additional relationship types can be identified and defined in the information model; they are modelled on the type level but will be created on the instance level in connection with creation of the entities. These structures enable creation of *dynamic client-server relationships* between pairs of entities. In each client-server connection, one entity - the client - demands services to be performed by the other entity - the server (figure 9).

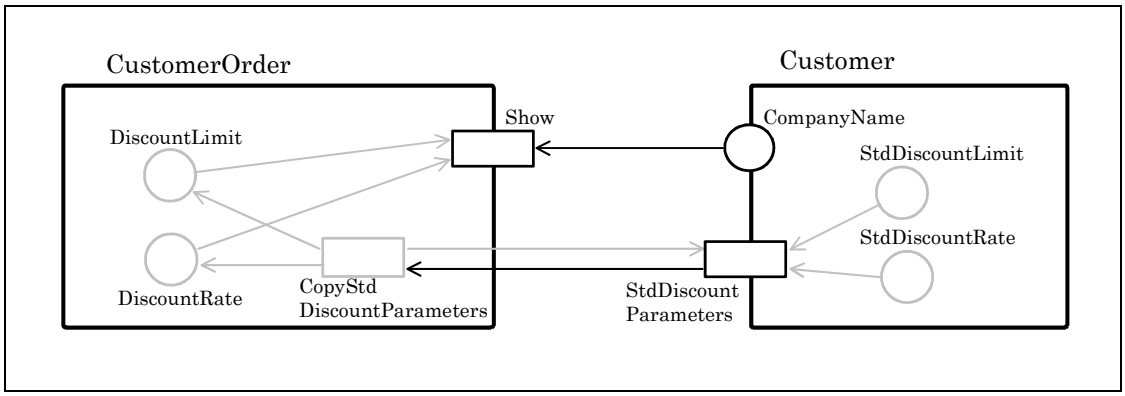

Figure 9 - A client-server relationship between two entity types

Three structuring concepts are defined: references and collections. A *reference* is a special attribute of which the value holds a direct link from one entity to another (represented as arrows between entity types of the taxonomy, see figure 10). Therefore, the main purpose of defining references is to handle *redundancy problems* and references are very simple and easily understood instruments for solving this because, by using references, the need for having multiple copies of data in different entities is eliminated. Because references are attributes, they are also defined by means of the classification abstraction mechanism.

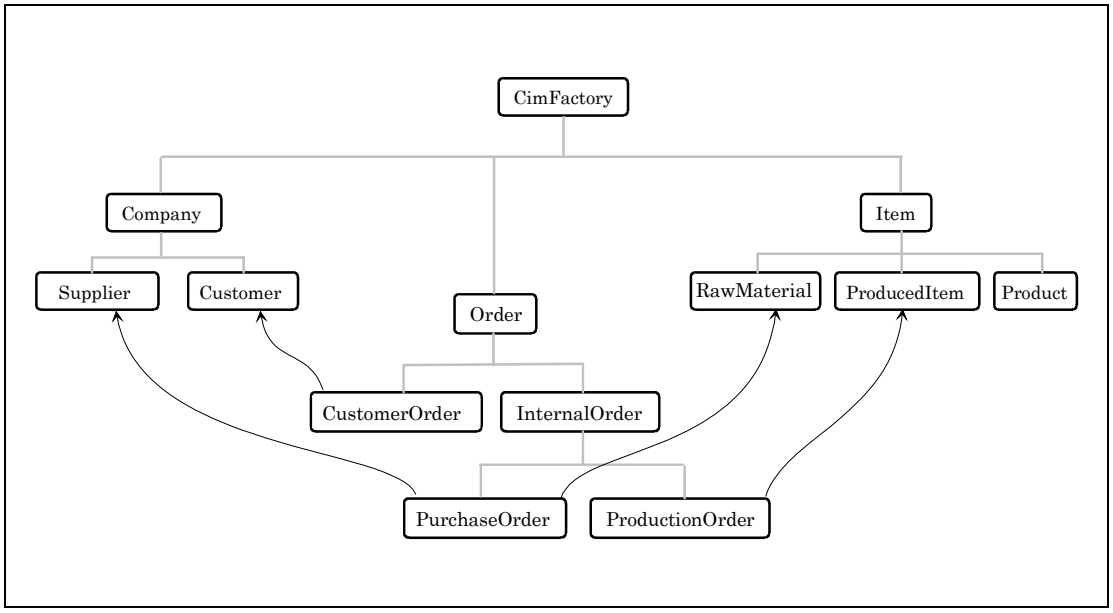

Figure 10 - References shown on top of the taxonomy

Like for the primary taxonomy, references can be defined in composite entity types referring to other entity types of the internal taxonomy but, in addition, references in the internal taxonomy can also refer to entity types outside the composite entity type (figure 11).

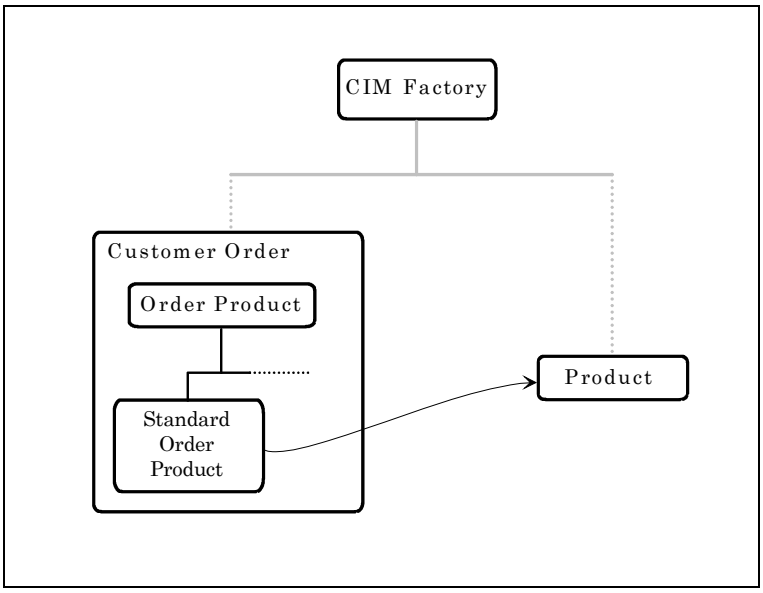

Figure 11 - A reference defined in an internal taxonomy

A *collection* is also defined as a directed relationship between two entity types, the *anchor* entity type and the *body* entity type. In the anchor entity type, each collection is a special attribute, which can also be identified by means of classification.

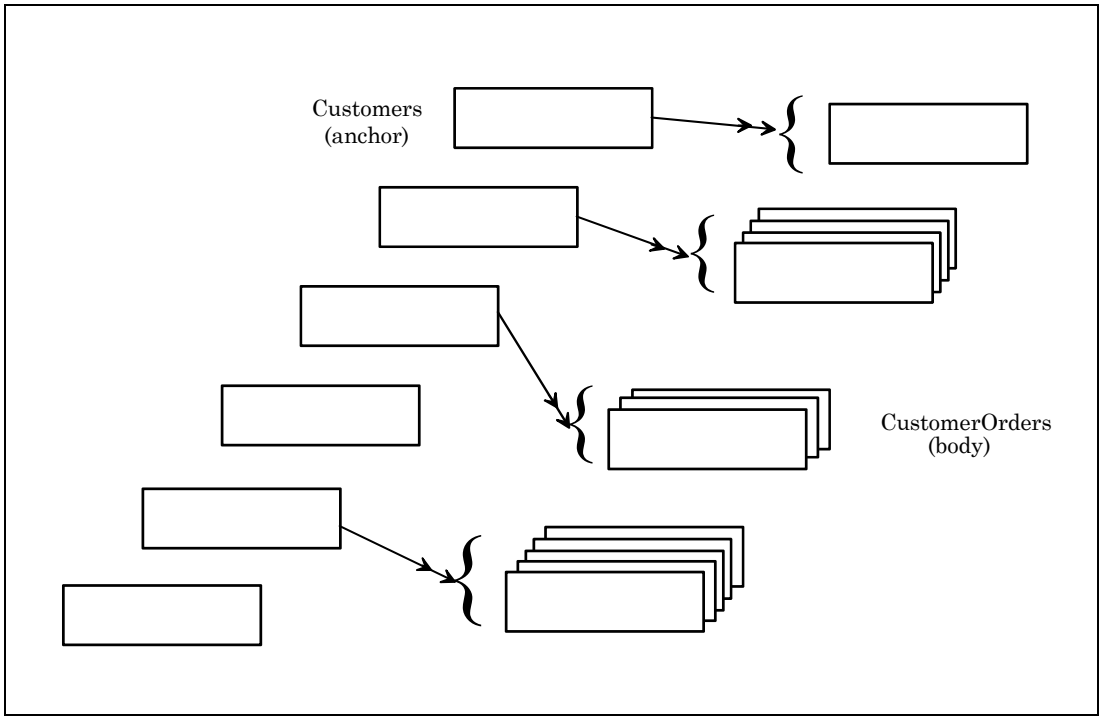

Figure 12 – Collections of entities, anchor and body entities

In each instance of a collection, one entity of the anchor type will exist while zero, one or more entities of the body type will exist (figure 12). From each anchor entity, the members of the body can be accessed as a whole, or individually. Hence, each collection defines an access path from instances of the anchor entity type to instances of the body entity types. To make such access paths as efficient as possible, information structures can be attached to collections. By these implementations, collections can be reduced to structures, which consist of only references.

The members of a collection are always accessible through a full scan operation but, to provide better access performance for certain retrieval operations, a sequential ordering of the collection can be

defined. Such an ordering can be specified by a search key and a structure efficient for search operations, e.g. a search tree.

In taxonomy diagrams, collections are represented as double-headed arrows (figure 13). Obviously, collections can be identified in the opposite direction of the references. However, many other collections can be defined (one example is shown in figure 13)

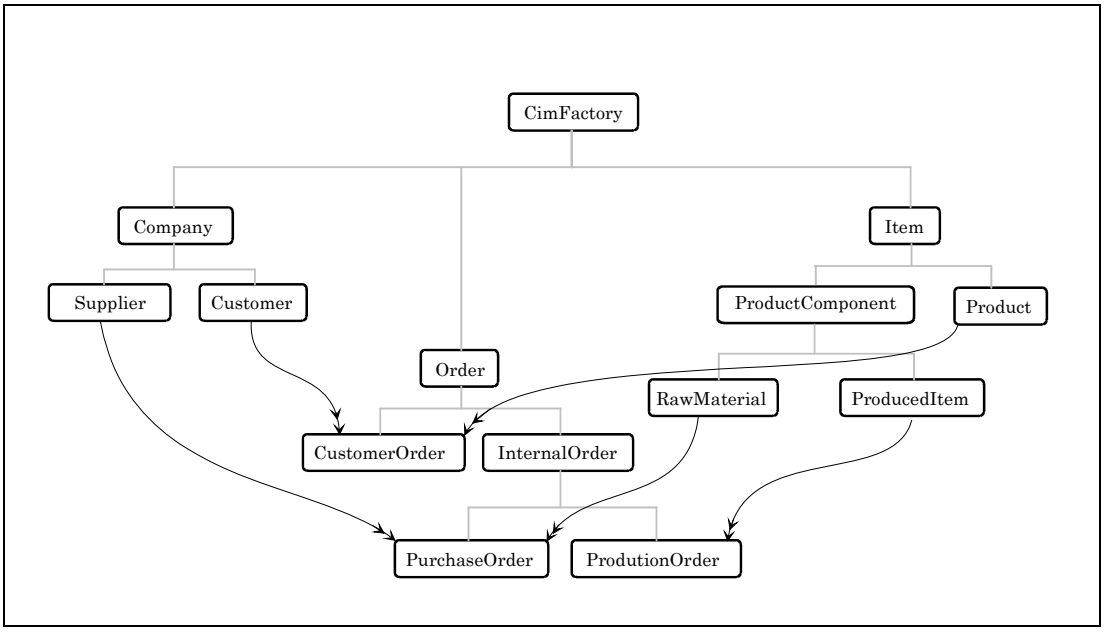

Figure 13 - Collections shown on top of the taxonomy

Collections are often termed *master-detail* relationships where the master type is the anchor type and the detail type is the body type. The definition of collections implies that composite entity types can alternatively be modelled by use of collections if the internal entity types are moved outside.

An *association* is a *bi-directional* relationship and, in the taxonomy, it can be indicated as a line with double arrowheads in both ends. Associations must always be reduced to collections, often by identification and definition of further entity types.

The *substitution* concept can be applied to references and collections. Further *semantics* of the information model can be attached to the relevant relationship types, e.g. *ordering*, *cardinality*, *composition*, *precedence constraints*, *referential integrity constraints* and *operational constraints*. Referential integrity constraints are attached to references while operational constraints, e.g. cascade or restricted delete are attached to collections. Cardinality is defined in relation to collections.

3 Fundamental Structures of the Information Model

As described in section 2, the entity types and relationship types of the information model are identified and determined by the abstraction mechanisms *classification* and *composition* and they form fundamental structures of the information model. Thus, two entity types are related to each other through classification or composition as illustrated in figure 14.

The primary relationship type of classification is the taxonomy, where each entity type, except the root type, has one parent type and each entity type can have a set of child types. If multiple inheritance is involved, some entity types can have multiple parent types. In the taxonomy, each parent type is a generalisation of its child types and each child type is a specialisation of its parent type. The relationships of composition are found in composite entity types, i.e. besides its attributes,

each composite entity includes an aggregation of all its internal entities and each internal entity is a separation from the composite entity it belongs to. As stated, the relationship types resulting from composition can be represented by collections. It is therefore remarkable that the classification abstraction mechanism can be used to identify all relationship types.

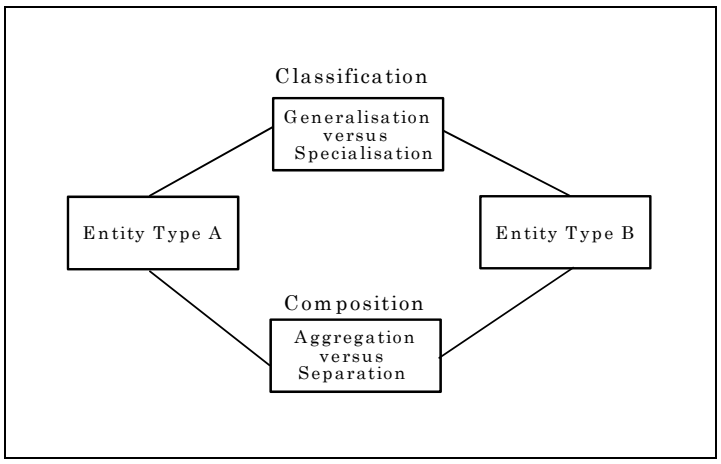

Figure 14 - Fundamental relationships between entity types

When a new entity type is added, it is necessary to consider how the type is related to other types by examining the classification and composition relationships. In this investigation, the position in the taxonomy will be determined, i.e. whether the entity type is attached to the main taxonomy or it is added as an internal entity type of a composite entity type. On the basis of its attributes, the new entity type is defined in relationship with its parent type and possible child types in the main taxonomy. Alternatively, possible composition relationships are considered by looking for entity types in which the new type can be added as an internal type (see figure 15).

The composition structure, which connects entity types with their possible internal entity types, the aggregation–separation structure, is a pure hierarchical structure. Similarly, the taxonomy, the generalisation–specialisation structure, is also a hierarchical structure, except when multiple inheritance exists.

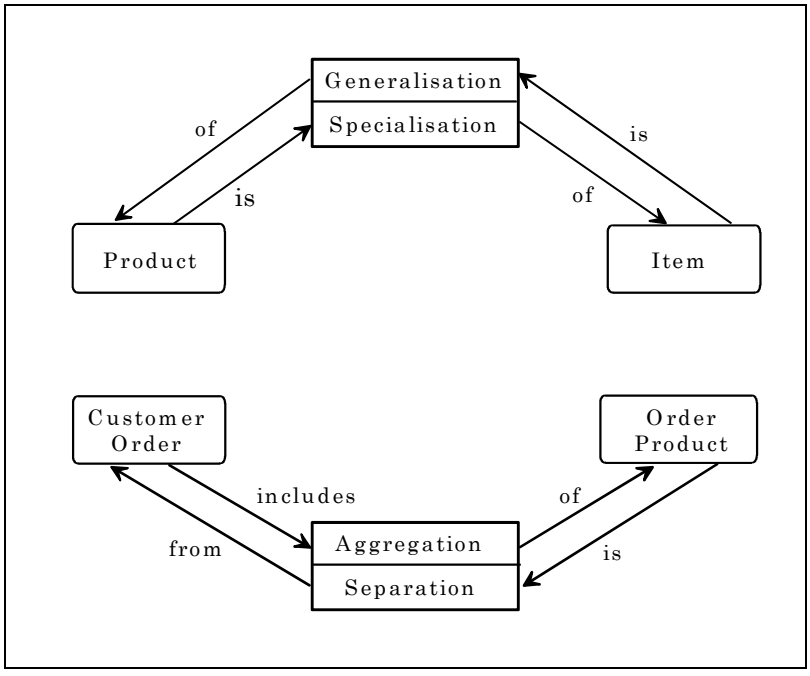

Figure 15 - Examples of relationship types

The remaining relationship types of the information model - *references* and *collections* - form cross-going structures in the taxonomy. References and collections define connections between entity types in the main taxonomy, between entity types in the internal taxonomies of composite entity types and between entity types in the main taxonomy and internal entity types of composite entity types.

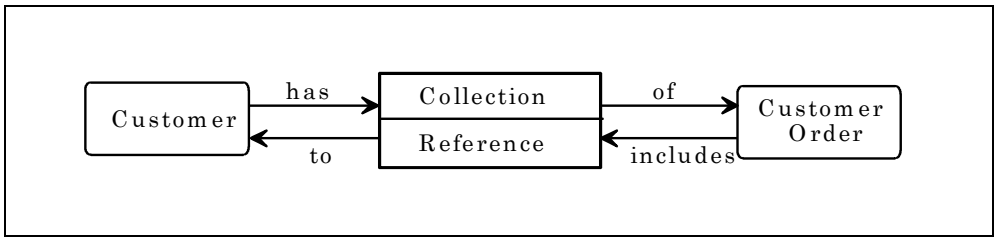

Figure 16 - An example of reference versus collection

As described in the previous section, internal structures, references and collections can be represented as special attributes of entity types; they are illustrated as a special symbol in object diagram (figure 4). This means that these attributes can be identified and defined by classification in the same way as ordinary attributes. In addition, the identification can be separated in time from further specification.

As stated, all relationship types are modelled on the type level of the information model. On the entity level, the different relationship types result in creation of different relationships. The structure of the taxonomy determines which attributes the entities contain when they are created.

References will cause direct links to be created. For each entity in which a reference is defined, the link to one particular entity must be established during the creation process. This implies that the referred entity must be created in advance.

For internal entity types, a reference to the entity type in which it is included, is automatically created, i.e. identical to the reference, which must be added, if the internal entity type is moved to the main taxonomy.

The implementation of collections is primarily a matter of how efficient it should be to retrieve the body entities to each anchor entity. Therefore, a collection does not necessarily cause specific links to be created on the entity level. In this case, retrieval operations will be based on the references in the body entities. If efficient retrieval operations are required, a particular structure over the body entities must be created for each anchor entity. More precisely, each node of the structure will refer to a particular body entity. As indicated, such structures can be materialised by means of references. Therefore, a reference is the most primitive relationship between two entities. This basic fact is one of the reasons why the traditional bi-directional relationship types - associations should - be divided into single-directional relationship types - references or collections.

4 The Information Modelling Methodology

Based on the definitions above, the information modelling methodology can be laid out. The modelling process consists of a specification of all the elements of the information model until a sufficient refinement has been achieved. In order to use the modelling resources as efficient and effectively as possible, the *most invariant specifications* should be defined first and the *least invariant* should be defined last. This process can be carried out by using various approaches and the activities can be performed sequentially or sometimes concurrently. However, some obvious restrictions exist; the entity types must be identified before any structure can be defined and semantics can be added. Based on this, it is recommended to divide the information modelling process into the following five *steps.*

A. Identification of entities and entity types.

As stated, the key components of an information model are the entity types. Therefore, in this step, the efforts are concentrated on identifying these types. To a great extent, an information model is a mapping of the real world. Therefore, it is natural to start looking for the basic entity types among the observed physical and logical objects in the physical system in which the information system is going to operate. This environment will be, e.g. a company, an organisation, a management system or a production system. For instance, the purpose of the information system is to store, maintain and share vital company information for members of the organisation or to support management functions of the company. In order for the information model to be application independent, the identification and specification of entity types should not be limited to what is strictly necessary for the required information system. Thus, a broader view of the foundation should be taken and reusability will be easier.

The concerned entities, and possible entity types, will often be called by names and, therefore, the first step should be to prepare a list of names of what is identified as important physical and logical real world objects to be modelled. During the information model design, these names would be evaluated and, if not excluded, they would possibly be defined in the model as entity types, entities, attributes of entities or internal entities of composite entity types. In the initial phase, it may not be clear what these names represent. Furthermore, an entity type should be identified as a template from which multiple entities will be created. An important categorisation of entity types is to analyse the frequency with which new entities of a type are created. Entity types with a low frequency must be identified and defined first because references will most often be made to these types.

B. Definition of sample attributes of the entity types.

The attributes, which should be included in the model, are dependent of the purpose of the information system. Thus, some evaluations and decisions have to be made in the modelling process about the use of the attributes. From the selected attributes, it is important to define the *identifier* of each type ([Copeland 1986]). Except for certain entity types, where only one entity exists, it is necessary to define such identifiers, in order to provide means for precise identification of information system objects. When identifiers are defined, it is also possible to identify further entity types or composite entity types by evaluating the functional dependencies between the attributes ([Codd 1979]).

C. Building taxonomies through classification.

In this step, the entity types are related to each other in a taxonomy by defining the most invariant relationships on the entity type level of the information model. Basically, this is performed by evaluating the two complementary operations of classification: generalisation and specialisation. General entity types are placed at the higher levels of the taxonomy and special entity types are placed at the lower levels. To find these relationships, further attributes are identified and the attributes of all entity types have to be examined for resemblance and differences. Two entity types, which have a sub-set of identical attributes, are related to each other in the taxonomy. If the attributes of one entity type are completely contained in the other, the first is a super-type to the second. If two entity types share common attributes but each has their own attributes, both of them will have the same ancestor-type, possibly the parent. If this does not already exist, it should be defined in the taxonomy. When the taxonomy has been arranged, it can be verified, in order to ensure that all possible entities can be described by the defined attributes. Also, it can be verified that no attributes have identical meanings.

D. Definition of relationship types between entity types of the taxonomy.

References represent fundamental relationships between pairs of entity types. Each value of a reference attribute establishes a direct access from one entity to another. References are defined in order to eliminate redundancy. If two entity types have attributes, which contain the

same piece of information, these attributes should be removed from one of the types and replaced by a reference.

As already defined, a collection is an oriented relationship between two entity types, the anchor type and the body type. Collections define master-detail relationships and establish access paths from each anchor entity to its body entities. These relationships can be used when user interfaces are created on the basis of the information model. Collections can be identified before references and subsequently reduced to structures containing only references.

Because the structures references and collections can be represented as attributes and determined by classification like all other attributes, they can be identified either in step C or D. This gives an important flexibility of the modelling process. However, if they are identified in step D, corresponding attributes should be added to the concerned entity types in the taxonomy.

E. Definition of semantics.

The semantic specifications are divided into two categories: attribute semantics and structural semantics. In the information model, the specifications are formulated in an explicit declarative form and, from the specified semantics, controlling operations can be created in the object base of the information system and in the user interfaces. Examples of attribute semantics are: required, uniqueness, initial values and validation constraints.

Identification of *composite entity types* is performed by reconsidering the steps A to D and their definition is carried out by performing the five steps recursively on each composite entity type.

A number of alternative approaches to the recommended approach can also be followed. For instance, the steps can be repeated in multiple cycles and, in each cycle, additional entity types can be identified and their attributes and structural positions can be defined. When attributes are defined, it is not necessary to define all attributes before the entity types can be classified. Enough attributes should be defined in order to perform generalisation versus specialisation in the classification process. Because specification of collections and semantics is primarily aimed at the user interfaces of the information system, these steps can be performed after the completion of the other steps. Composite entity types can be defined in concurrent tasks.

5 Information Modelling Based on Abstraction Mechanisms

It is a rather unique quality of this research work that a consistent set of abstraction mechanisms is formed as a precise *science paradigm* and foundation for the new methodology. Originally, only generalisation and aggregation have been considered sub-mechanisms ([Smith 1977a] and [Smith 1977b]) and the importance of using the complementary abstraction mechanisms of the two has not been much underlined since, i.e. specialisation in contrast to generalisation and separation in contrast to aggregation. In fact, it is essential for the use of classification and composition that the abstraction processes are performed as two-way considerations.

In other methodologies ([Booch 1986], [Booch 1994], [Coad 1990], [Coad 1991], [Rumbaugh 1991] and [Jacobson 1992]), which have adopted the generalisation abstraction mechanism, the taxonomy does not include the entire set of entity types of the model. Instead, a number of local taxonomies can be defined in information models. The complete taxonomy with the entire set of entity types has not been introduced before as the basis of the information model and as the "back-bone" for definition of composition structures. Collecting all entity types in one total taxonomy emphasises that identification and definition of entity types and attributes must be performed closely related to the complete set of entity types and the structure of the taxonomy. In the taxonomy, a number of entity types are defined even though entities of these types will not be generated, i.e. many super types. Such entity types are not only important for definition of the taxonomy; they can be the foundation for making many compositional decisions in the modelling process.

In addition to modelling one total taxonomy of the information model, it is defined that entity types can be regarded as composite entity types. Thereby, the methodology offers the opportunity to define entity types in a recursive way and each composite entity type can be regarded as an origin for an additional world which is defined by creating an internal information model - based on an internal taxonomy.

It is an important aspect of the methodology to state both classification and composition as a united foundation of the central characteristics of object-orientation. Many previous attempts to define object-orientation have lead to confusing definitions because these have focused on inheritance and client-server relationships as two separate characteristics. Seen in the perspective of the new methodology, these definitions can be regarded as two different aspects of object-orientation. Thus, inheritance is merely an aspect of the classification abstraction mechanism and client-server relationships are clearly seen as an architectural perspective of the composition abstraction mechanism.

6 Entity Relationship Modelling in a New Perspective

In existing entity-relationship (ER) techniques ([Chen 1976], [Rock-Evans 1987] and [Rumbaugh 1991]), relationships are defined as binary connections, which represent many different elements of structural information. In this proposal, the different structures are modelled separately. In this way, they can be expressed more clearly. The proposed methodology is an information modelling approach because the entity types are modelled primarily from an *information point of view*. Thereby, the methodology represents a shift of paradigm. Data is regarded as the primary characteristics of entity types and a greater attention is paid to identification of entity types and their attributes as the most invariant components and structures. The ER techniques do not provide means for separate and independent identification of entity types. From a *function point of view*, the entity types are modelled with a set of services, which can perform specific operations in connection with the data attributes. Thereby, a significant part of the functionality of a system is developed in an *application independent form* and traditional function modelling can be reduced to development of *user interfaces.*

All the relationships of an information model can be identified and defined in connection with the taxonomy. Therefore, the taxonomy can be regarded as a kind of "back-bone" structure from which further modelling can be performed. In order to benefit from the advantages of graphical representations of information models, graphic-oriented development tools can be developed in relation to the methodology. Such a tool system could be a valuable support for the participants of information modelling projects. The tool should provide assistance for creation, manipulation and communication of the different kinds of graphs, which are the foundation for the development process. The specifications of entity types should be stored in the system from which the participants should be able to retrieve them and add further details to the specifications according to the stepwise refinement approach.

7 Creation of Relational Data Models

The information modelling methodology supports some of the most important design tasks, independent of the data model being used for implementation. Naturally, the information model will be most suitable for the implementation as an object oriented database system. Furthermore, the methodology also conforms to the *object-relational data model* ([Stonebraker 1996]) but, as the relational data model is the most widely known data model, it is important to consider how the proposed information model can be a foundation for creation of relational data model.

In an entity type taxonomy, all the basic entity types are defined, each with a static relationship to other types. A table (relation) of a relational data model ([Codd 1970] and [Codd 1979]) is comparable with an entity type with only data attributes and no methods. Consequently, a row (tuple) of a table is comparable with an entity of an entity type. The table attributes are comparable with the data attributes of the entity type. If it is possible to specify stored procedures in the database, these

procedures can be derived from the methods of entity types. Specification of constraints is possible in newer versions of the relational data model.

When the information model is implemented as a relational data model, some design choices will have to be made in order to derive table definitions from the entity types. The primary criterion for deriving tables from the taxonomy is the placement of identifiers. Each entity type, in which an identifier is defined, should be created as a table with the identifier as the *primary key.* Each sub-type of an identifier type can also be created as a table and there will be a one to one relationship between rows of such a pair of tables. Therefore, in order to link the rows, the identifier must be copied to all the sub-tables as the primary key, which then also will be a *foreign key.* Thus, corresponding rows must have the same value of the identifier.

If sub-types of an identifier type are not created as individual tables, they can be combined into the super-table. In this table, a special column should be added as a case attribute, indicating the sub-type for each row. As a result, a number of columns will have null values in the rows of the table. In each row, only the columns of one sub-type will have non-null values assigned. Special considerations have to be made about super-types to the type in which identifiers are defined. If a super-type is without data attributes, it will not be necessary to create a table for such an entity type. Otherwise, any data attributes of the super-types must be duplicated in each identifier-table.

As indicated, the basic part of an object-oriented information model, the taxonomy, and highlights some key decisions for the design of the database. Thus, among the total number of logical relationships between tables of the database, some of them have the special and separate meaning, which is well defined by the taxonomy. The structures, which are defined by references, have to be treated in the same way in the relational data model as in the proposed information model. They are created in the tables as foreign keys and each of them contains the value of the primary key of the particular row in the table, which it refers to. The relational data model provides no means for implementation of collections. They can be created dynamically as join operations based on foreign keys. This means that enforcement of the constraints, which are defined in connection with collections, will have to be specified in connection with foreign keys.

8 Comparison with other Methodologies

Most available methodologies are based on the *entity relationship* technique, which was introduced by Chen in 1976 ([Chen 1976]) or comparable techniques ([Rock-Evans 1987] and [Martin 1990]). This technique has not changed significantly since it was developed, and a number of disadvantages have been found with respect to invariance and reusability. Compared to these ER techniques, a reference is a very simple *structural mechanism*. ER techniques are used for both the identification of entity types and for definition of relationships. The identification is primarily performed on the basis of relationships and not on the basis of an abstraction mechanism, which focuses on attributes. Moreover, with ER not all entity types are identified because relationship types can include attributes so that they are actually not relationships but entity types. A total structure between entity types like the taxonomy showing the attribute dependencies is not provided. Some newer methodologies have introduced the possibility to create inheritance relationships between a subset of entity types ([Coad 1991], [Rumbaugh 1991], [Jacobson 1992], [Bancilhon 1992] and [UML???]).

In the presented methodology, the identification and specification of entity types results in a relationship structure on the type level, the taxonomy, on which additional relationships of the information model are modelled and, because attributes are identified and defined only in entity types, a clear distinction is made between entity types and relationship types. In contrast to traditional ER techniques, the identification and definition of the different relationship types are of secondary importance.

The introduction of *composite entity types* adds another dimension to the modelling approach. It provides the designers an extra degree of flexibility in relation to stepwise refinement. The model can be build in steps by adding further details both to the primary taxonomy and to the internal taxonomies of composite entity types. Defining composite entity types by creating an internal

taxonomy is a natural extension of the idea that the basic structure of an information model is the taxonomy structure. Thereby, the methodology is characterised as a recursive methodology. Each composite entity type is regarded as an origin from which an internal information model is developed.

The object base of the information system can be generated directly from the information model and the user interfaces can be generated on the basis of both the information model and the function model. However, it is a unique implication of information modelling that a large part of user interfaces can be extracted only from the information model, i.e. the services, which are defined with the entity types of the model. The function model is primarily created in order to identify user interfaces and to define, which entity types are used by which user interfaces.

9 Conclusion

Information modelling is important in order to focus on a data driven approach for development of such information systems, which are relatively invariant against changes in functional requirements. Therefore, by building an information model, the fundamental design decisions regarding reusability, extensibility and reliability can be made in a structured way and the information model can serve as the foundation for construction of the resulting information system. The presented information modelling methodology is based on these requirements.

The methodology is based on a defined science paradigm with a set of fundamental abstraction mechanisms and on the object-oriented paradigm, which is applied in harmony with the abstraction mechanisms. Two abstraction mechanisms - classification and composition - are defined on the basis of the fundamental concepts analysis and synthesis. These mechanisms provide the means to set particular focus on decisions related to the type level as the most invariant decisions. On this level of the information model, the basic components, the entity types, are identified and defined primarily by means of the classification abstraction mechanism and the entity types are organised in a taxonomy. In addition, it is defined that every entity type can be a composite entity type, which can be regarded as a frame for an internal information model. The introduction of composite entity types adds another dimension to the modelling approach and enables recursive modelling of multi level complexity.

The different relationship types are defined as special attributes, which can be identified and defined the same way as other attributes in the taxonomy by use of classification. This emphasises that the primary importance is identification of entity types and that identification of relationships are of secondary importance. Compared to most entity-relationship (ER) techniques, references are pure structural mechanisms and sufficient means to handle redundancy problems. By defining the key characteristics of attributes in entity types, a clear distinction is made between entity types and relationship types. Furthermore, the different relationship types are modelled separately and thereby expressed more clearly.

The addressed key issues of the information modelling methodology have impact on analysis and synthesis of information systems. When an existing information system is analysed, the system documentation of such an analysis should be build on the basis of an information model. Such a model should be organised with respect to invariance. When a new information system is developed, an information model should be designed by following the guidelines of the presented modelling methodology. Decisions about the most invariant components and structures should be made first and the remaining decisions should be made relatively.

References

[Avison 1988]
D. E. Avison, G. Fitzgerald: *Information Systems Development: Methodologies, Techniques and Tools.* Blackwell Scientific Publications, 1988.

[Bancilhon 1992]
Francois Bancilhon: Understanding Object-Oriented Database Systems. In Proceedings of the 3rd International Conference on Extending Database Technology (EDBT '92), Vienna, 23-27 March 1992. Springer-Verlag, Berlin.

[Bertalanffy 1967]
Ludwig von Bertalanffy: *General Theory of Systems - Application to Psychology*. Social Science Information, Vol. 6, 1967, pp.125-136.

[Booch 1986]
G. Booch: *Object-Oriented Development.* IEEE Transactions on Software Engineering, Vol. SE-12, no. 12, Dec. 1986.

[Booch 1994]
G. Booch: Object-Oriented Analysis and Design With Applications. Second edition. Benjamin/Cummings, Redwood City, California, 1994.

[Boulding 1956]
K. Boulding: *General Systems Theory - The Skeleton of Science.* General Systems Vol.1, 1956, p.11.

[Chen 1976]
P. Chen: *The Entity-Relationship Model - Toward a Unified View of Data.* ACM Transactions on Database Systems Vol. 1, pp.9-36 1976.

[Coad 1990]
P. Coad, E. Yourdon: *Object-Oriented Analysis.* Yourdon Press, 1990.

[Coad 1991]
P. Coad, E. Yourdon: *Object-Oriented Design.* Yourdon Press, 1991.

[Codd 1970]
E.F. Codd: *A Relational Model of Data for Large Shared Data Banks.* Communications of the ACM, Vol. 13, no. 6. pp.377-387 1970.

[Codd 1979]
E.F. Codd: *Extending the Database Relational Model to Capture More Meaning.* ACM Transactions on Database Systems, Vol. 4, no. 4. pp.397-434 1979.

[Copeland 1986]
G. Copeland, S. Khoshafian: *Object Identity.* Proceedings on OOPSLA, Portland, Oregon USA. 1986.

[Hammer 1978]
Michael Hammer and Dennis McLeod: *The Semantic Data Model: A Modelling Mechanism for Data Base Applications.* Proceedings of ACM/SIGMOD International Conference on Management of Data. Austin Texas, pp.144-156, 1978.

[Jacobson 1992]
Ivar Jacobson: *Object-Oriented Software Engineering.* Addison-Wesley, Reading, Mass. 1992.

[Martin 1990]
James Martin: *Information Engineering.* Prentice Hall, New Jersey, 1990.

[Rock-Evans 1987]
Rosemary Rock-Evans: *Analysis Within The Systems Life Cycle.* Pergamon Infotech Limited, United Kingdom, 1987.

[Rosch 1978]
Eleanor Rosch: *Principles of Categorisation.* In: Cognition and Categorization. Laurence Erlbaum, Hillsdale, Jew Jersey, 1978.

[Rumbaugh 1991]
James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen: *Object-Oriented Modelling and Design.* Prentice Hall, New Jersey, 1991.

[Smith 1977a]
J. M. Smith, D. C. P. Smith: *Database Abstractions: Aggregation.* Communications of the ACM vol.20, no.6. pp.405-413 New York 1977.

[Smith 1977b]

J. M. Smith, D. C. P. Smith: *Database Abstractions: Aggregation and Generalization.* ACM transactions on Data Base Systems, vol.2, no.2. pp.105-133 New York 1977.

[Sowa 1984]

John F. Sowa: *Conceptual Structures: Information Processing in Mind and Machine.* Addison-Wesley, 1984.

[Stonebraker 1996]

Michael Stonebraker: Object-Relational DBMSs. Morgan Kaufmann Publishers, San Francisco 1996.